Integrated Airgap Insertion and Layer Reassignment for Circuit Timing Optimization

Younggwang Jung, Daijoon Hyun, and Youngsoo Shin
School of Electrical Engineering, KAIST
Daejeon 34141, Korea

***Abstract*—Airgap is an intentional void formed in inter-metal dielectric (IMD). It brings about reduced coupling capacitance, and so can be used to improve circuit timing. Airgap can be utilized in a limited number of metal layers due to its high process cost. For given airgap layers, two problems should be addressed to insert airgap: relocate some metal segments in non-airgap layers into airgap layers (called layer reassignment) and determine the amount of airgap for each metal segment in airgap layers (airgap insertion). Two problems are solved together in this paper with a goal of maximizing setup total negative slack (TNS) while assuring no hold violations. It is formulated as mixed integer quadratically constrained programming (MIQCP); heuristic algorithm is proposed for practical application and its performance against MIQCP is experimentally assessed using small test circuits. Experiments demonstrate that TNS and WNS are improved by 35% and 10%, respectively, while simple minded approach achieves 6% and 4% less improvements compared to the proposed method.**

I. Introduction

Airgap is an intended void formed in some dielectric material, used for inter-metal dielectric (IMD). It reduces IMD permittivity from 2.4 (porous SiCOH) to 2.0 or lower (airgap with SiCOH) [1]–[3] and causes reduction in coupling capacitance. It is demonstrated that there is 20% of reduction in total wire capacitance through airgap formation [1]. Airgap formation is a costly process including an extra mask and more than 10 additional process steps [4], [5]. So the number of layers that accommodate airgaps, called an airgap layer in this paper, is practically limited, for example 2 layers [6], [7]. Airgaps can be used to improve circuit timing thanks to their reduced capacitance. The two problems can be considered in this respect. For wire segments in airgap layers, the problem is to determine the amount of their airgaps (this problem will be called airgap insertion). For those not in airgap layers, they may be reassigned to one of airgap layers (called layer reassignment problem), which is then followed by airgap insertion. Previously, two problems have been solved one by one (layer reassignment followed by airgap insertion) with goal of optimizing setup timing margins [8]. This approach is less effective for further timing optimization and there is no consideration for hold time constraints. In this paper, the two problems are solved together while both setup and hold timings are considered.

A. Motivational Example and Problem Description

Assume a simple net shown in Fig. 1(a), which consists of five wire segments with their metal layers shown in parentheses. Let us consider the hold slack SLK_1^{hd} of a timing path p_1 and the setup slack SLK_2^{su} of p_2. Let M3 and M4 be airgap layers. One simple solution of layer reassignment and airgap insertion is shown in Fig. 1(b), in which we assume that the goal is to remove negative setup slack (i.e. to make SLK_2^{su} non-negative). Since w_1 is already in airgap layer M4, the maximum amount of airgap is inserted; w_3 and w_4 are reassigned to airgap layers followed by airgap insertion; it is assumed that w_2 cannot be reassigned to M4 due to conflict; w_5 can be reassigned to M4 but airgap is not inserted. The airgap insertion of each wire is associated with two delay changes (x, y), which affect SLK_1^{hd} and SLK_2^{su}, respectively. The x and y include the delay component of a gate and interconnect to each fanout gate; they usually have different values due to different input transition times used for hold and setup timing analysis as well as different interconnect delays to fanouts. As shown in the figure, this simple approach achieves the goal but SLK_1^{hd} becomes negative. The best solution with unified layer reassignment and airgap insertion is shown in Fig. 1(c), in which the goal is to make hold slack non-negative and maximize the setup slack.

In this paper, we address a problem of airgap insertion with layer reassignment, taking into account both setup and hold constraints. An exact solution is obtained by formulating the problem as a mixed-integer quadratically constrained programming (MIQCP). A heuristic algorithm is also proposed for practical application. We first perform layer reassignment with setup constraint for the nets, called setup critical nets, on setup timing critical paths, in which the wires on airgap layers are assumed to be filled with airgap, i.e. airgap insertion problem is not solved. After timing update, hold critical nets whose delay reductions cause hold time violations are identified, and their optimizations are restored. For each hold critical net, we perform airgap insertion with layer reassignment taking into account both setup and hold constraints, in which solution candidates are repeatedly searched in the decreasing order of setup total negative slack (TNS) improvement and the first candidate that satisfies hold constraints is selected as a solution.

The remainder of this paper is organized as follows. In Section II, the problem is defined and formulated through MIQCP. Section III presents a heuristic algorithm for practical runtime. Experimental results are demonstrated in Section IV, in which we show the effectiveness of proposed method in terms of timing parameters and runtime. Conclusions are drawn in Section V.

978-1-7281-4123-7/20/$31.00 © 2020 IEEE

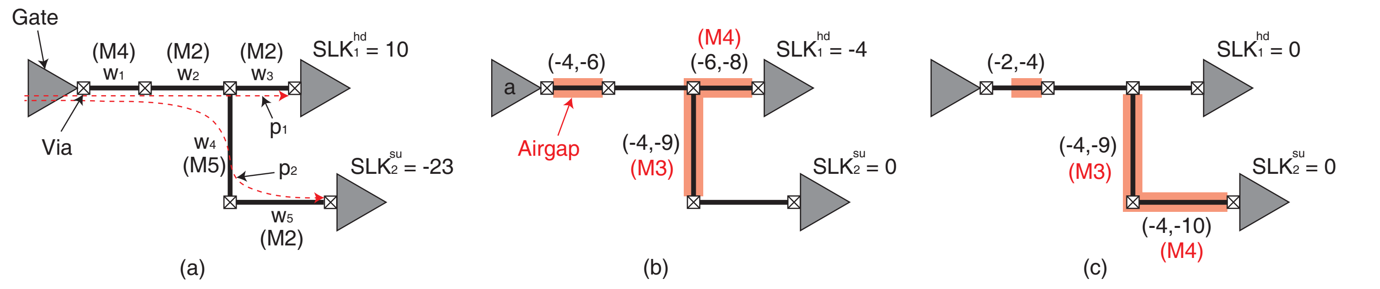


Fig. 1. A motivational example: (a) initial net with five wire segments that are assigned to the layers within parentheses, (b) simple solution of layer reassignment and airgap insertion, and (c) the best solution with unified layer reassignment and airgap insertion.

II. Airgap Insertion with Layer Reassignment

A. Problem Definition

Let a post-routing netlist be given. Metal layers where airgap can be inserted are fixed. Full dummy fill is assumed, so the amount of airgap that can be inserted for a particular wire segment is independent of where it is located. For a wire w_i, the maximum amount of airgap that can be inserted is identified, which allows us to associate w_i with the range of wire capacitance $C_{i,l}$ when the wire is assigned to layer l.

The problem of airgap insertion with layer reassignment is to determine the layer of each setup critical wire w_i, which constitutes each setup critical net, and its capacitance $C_{i,l}$, if the wire is assigned to layer l, such that setup TNS is maximized and all hold constraints are satisfied.

B. Exact Solution through MIQCP

The problem is formulated as MIQCP to get an exact solution (for small circuits) as well as to gain a better intuition. A heuristic algorithm is presented in Section III.

Each wire w_i is associated with a binary variable $x_{i,l}$, which is 1 if w_i is assigned to layer l and 0 otherwise. It is assigned to only one metal layer, which is ensured by

$$\sum_{l\in L} x_{i,l} = 1, \tag{1}$$

where L is a set of metal layers to which each wire can be assigned. As reserved routing is assumed, wires cannot move to the layers with different direction. For instance, a horizontal wire w_i in M2 is allowed to move into M4 and M6, whose direction is horizontal; this is ensured that there are no variables, $x_{i,3}$ and $x_{i,5}$.

Consider Fig. 2(a), where w_i in M2 is connected with w_j in M3, and there is another wire w_k in M4. w_i cannot move to M4 due to electrical short with w_k, as illustrated in Fig. 2(b), called wire conflict and this is avoided by

$$x_{i,l} + x_{k,l} \le 1, \tag{2}$$

where w_i and w_k are the pair of wires that pass through a common area. There is no wire in M6, so w_i seems to be allowed to move into M6. But it causes the overlap of w_k and the stacked via, which connects w_i to w_j, as illustrated in Fig. 2(c), called via conflict and it is prevented by

$$x_{i,l} + x_{j,l'} + x_{k,l''} \le 2, \tag{3}$$

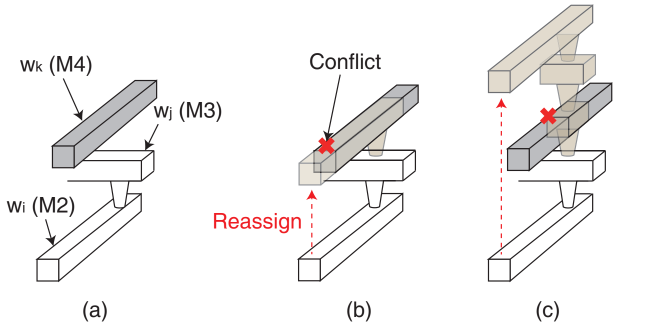


Fig. 2. (a) Example of post-route metal layout, (b) wire conflict when w_i is reassigned to M4, and (c) via conflict when moving w_i to M6.

where w_i and w_j are connected by vias, w_k passes through the via area, and l'' is the layer between l and l'; e.g. $(l, l', l'') = \{(2, 4, 3), (6, 3, 5), ...\}$.

Wire resistance R_i of w_i depends on the assigned layers of w_i and w_j, which is the adjacent upstream wire of w_i, because R_i includes the resistance of vias that connect the wires, which is ensured by

$$R_i = \sum_{l\in L}\sum_{l'\in L} R_{i,l}^{j,l'}\, x_{i,l}\, x_{j,l'}, \tag{4}$$

in which $R_{i,l}^{j,l'}$ is the sum of wire resistance of w_i and the via resistance when w_i and w_j are assigned to l and l', respectively. Wire capacitance $C_{i,l}$ of w_i at layer l is fixed to a constant $C_{i,l}^{max}$, if the layer is not in airgap layers. But for l belonging to airgap layers, $C_{i,l}$ exists between $C_{i,l}^{min}$ and $C_{i,l}^{max}$; that is, $C_{i,l}$ is a continuous variable representing the amount of airgap, which is given by

$$\sum_{l\in L_{ag}} x_{i,l}C_{i,l} + G(1 - \sum_{l\in L_{ag}} x_{i,l}) \ge C_{i,l}^{min}, \tag{5}$$

$$\sum_{l\in L_{ag}} x_{i,l}C_{i,l} - G(1 - \sum_{l\in L_{ag}} x_{i,l}) \le C_{i,l}^{max}, \tag{6}$$

$$G\sum_{l\in L_{ag}} x_{i,l} + (1 - \sum_{l\in L_{ag}} x_{i,l})C_{i,l} \ge C_{i,l}^{max}, \tag{7}$$

$$-G\sum_{l\in L_{ag}} x_{i,l} + (1 - \sum_{l\in L_{ag}} x_{i,l})C_{i,l} \le C_{i,l}^{max}, \tag{8}$$

where L_{ag} represents a set of airgap layers. C_i is determined by the capacitance variable of the assigned layer of w_i, as listed in

$$C_i = \sum_{l \in L} C_{i,l}\, x_{i,l}. \tag{9}$$

Gate delay changes for setup and hold analyses, $\Delta d^{su}_{g,n}$ and $\Delta d^{hd}_{g,n}$, are linearly modeled by the change of load capacitance, ΔC_n, in which load capacitance is the sum of all wire capacitances included in net n and its change is the difference from initial value.

$$\Delta d^{su}_{g,n} = s^{su}_n\, \Delta C_n, \tag{10}$$

$$\Delta d^{hd}_{g,n} = s^{hd}_n\, \Delta C_n. \tag{11}$$

Wire delay change is the difference from initial wire delay $\hat{d}_{w,m}$ calculated by Elmore delay model [9], which is given by

$$\Delta d_{w,m} = \sum_{i \in W_m} \sum_{j \in W_{d_i}} R_i\, C_j \,-\, \hat{d}_{w,m}, \tag{12}$$

where $\Delta d_{w,m}$ is the wire delay change for a timing path m in interconnect, W_m is a set of wires on m, and W_{d_i} is a set of downstream wires of w_i. Nonlinear product of three binary variables, caused by $R_i C_j$ in constraint (12), is substituted to a binary variable by using linearization technique [8].

The delay change of setup critical path p, Δd^{su}_p, is the sum of gate delay changes and wire delay changes in the path, and its reduction increases setup slack of the path. Setup WNS at endpoint e, WNS^{su}_e, is the worst negative slack of the paths to e.

$$\Delta d^{su}_p = \sum_{n \in N_p} \Delta d^{su}_{g,n} + \sum_{m \in M_p} \Delta d_{w,m}, \tag{13}$$

$$WNS^{su}_e \le S\hat{L}K^{su}_p - \Delta d^{su}_p, \tag{14}$$

$$WNS^{su}_e \le 0, \tag{15}$$

where $S\hat{L}K^{su}_p$ is initial setup slack of path p. The delay reduction in setup critical net may cause hold violation, when the net is included in hold critical path. Hold slack is prevented from being a negative value by

$$\Delta d^{hd}_p = \sum_{n \in N_p} \Delta d^{hd}_{g,n} + \sum_{m \in M_p} \Delta d_{w,m}, \tag{16}$$

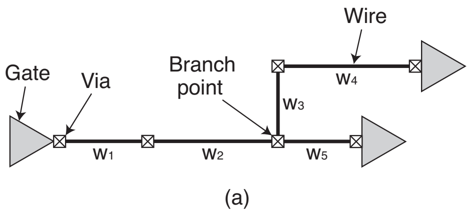
$$S\hat{L}K^{hd}_p + \Delta d^{hd}_p \ge 0, \tag{17}$$

where $S\hat{L}K^{hd}_p$ is initial hold slack of path p.

Airgap insertion with layer reassignment can now be defined as the problem to determine the layer and the wire capacitance of each setup critical wire, with the objective of maximizing setup TNS. TNS is defined by the sum of setup WNS in all endpoints, so the objective is given by

$$\text{Maximize} \quad \sum_{e \in E} WNS^{su}_e, \tag{18}$$

where E is a set of endpoints.



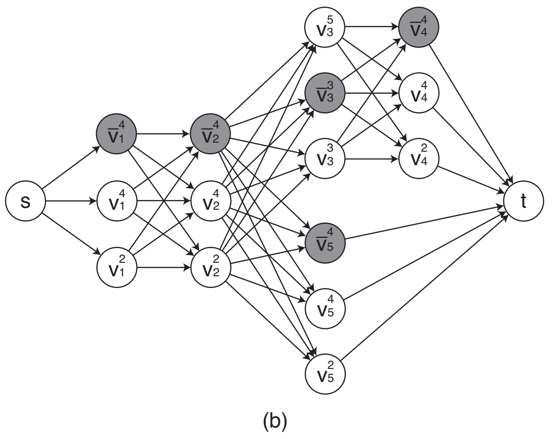


Fig. 3. (a) A net with five wire segments and (b) its corresponding directed acyclic graph, when assuming M3 and M4 as airgap layers.

III. Heuristic method

A. Overall Flow

For a given post-route circuit, we first identify setup critical nets on timing critical paths with negative setup slack. Each net is then modeled with a directed acyclic graph, where each wire-segment in a net is modeled by multiple vertices that represent the layer each of which a wire-segment is assigned to and whether airgap is inserted in a wire-segment. Edges connect the vertices of adjacent wires, and each edge is associated with a weight that is infinity if its directed vertex corresponds to wire conflict or via conflict; corresponding reduction of setup TNS, otherwise. Layer reassignment with setup constraint is performed for the graphs, where we visit the graph of the net with the most setup critical paths first. We find a subgraph with the minimum sum of edge weights; this corresponds to the results with the largest increase in setup TNS.

After timing update, hold critical nets that cause hold violations in layer reassignment are identified, and we calculate the value of increased setup slack over hold slack reduction caused by the delay reduction of each net; this is used as a measure of how much setup slack can be improved by reducing a certain amount of hold slack. Airgap insertion with layer reassignment considering both setup and hold constraints is performed for each hold critical net, in the decreasing order of the measure. In the same graph used in layer reassignment, we repeatedly find subgraphs in the increasing order of the sum of edge weights, and the first subgraph without hold violation is selected as a solution. For every net optimization, timing information is updated and some nets with worst hold slack less than 0.1ps are skipped.

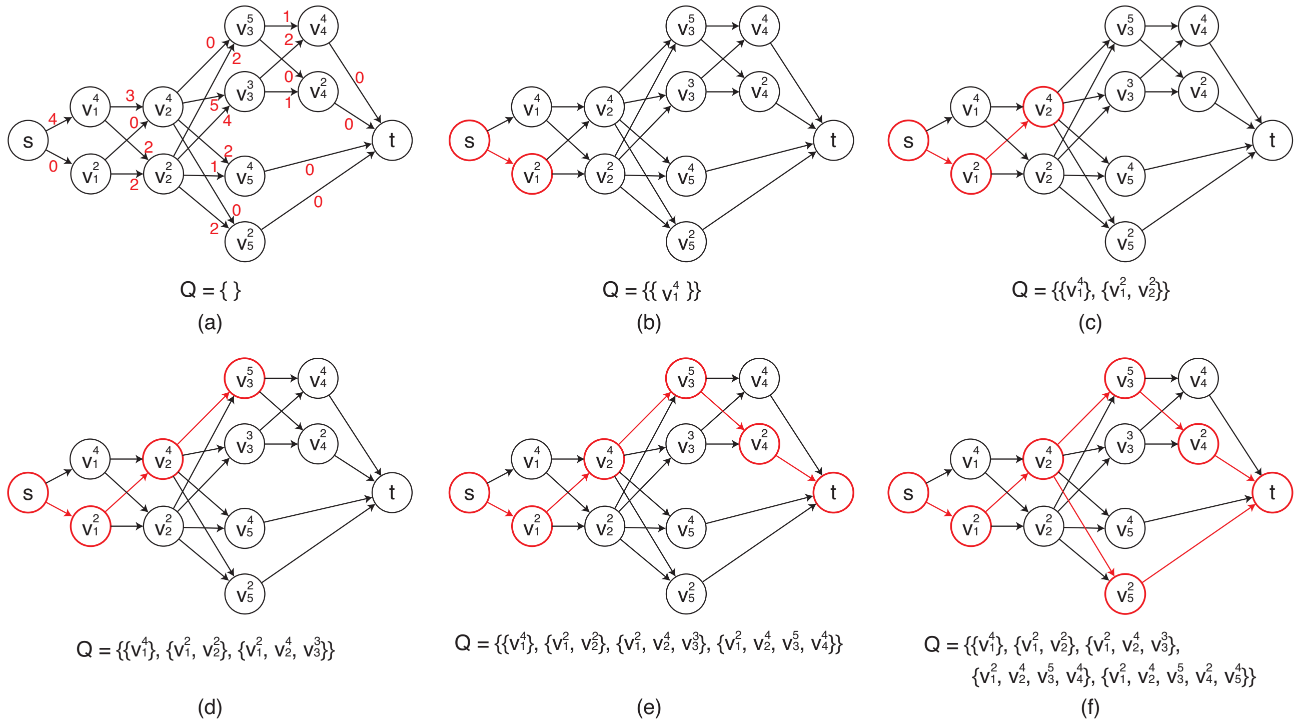

s
t
v_1^4
v_1^2
v_2^4
v_2^2
v_3^5
v_3^3
v_4^4
v_4^2
v_5^4
v_5^2
Q = { }
(a)
Q = {{ v_1^4 }}
(b)
Q = {{v_1^4}, {v_1^2, v_2^2}}
(c)
Q = {{v_1^4}, {v_1^2, v_2^2}, {v_1^2, v_2^4, v_3^3}}
(d)
Q = {{v_1^4}, {v_1^2, v_2^2}, {v_1^2, v_2^4, v_3^3}, {v_1^2, v_2^4, v_3^5, v_4^4}}
(e)
Q = {{v_1^4}, {v_1^2, v_2^2}, {v_1^2, v_2^4, v_3^3},
{v_1^2, v_2^4, v_3^5, v_4^4}, {v_1^2, v_2^4, v_3^5, v_4^2, v_5^4}}
(f)

Fig. 4. An example of AILR algorithm in finding the first subgraph with the minimum sum of edge weights.

B. Graph Modeling

Consider Fig. 3(a), in which a net consists of several wire segments, and adjacent wires in different layers are connected by via. The point at which one path branches to multiple fanout gates is called branch point, and branch is defined as a set of wires between branch points or between branch point and gate pin; e.g. $\{w_1, w_2\}$, $\{w_3, w_4\}$, and $\{w_5\}$. The net is modeled with a directed acyclic graph $G = (V_e \cup V_f, E)$ in Fig. 3(b), where a vertex v_i^l in V_e, represented in white circle, indicates that a wire w_i is assigned to l-th layer without airgap. A vertex $\overline{v}_i^l$ in V_f, marked with gray circle, represents that w_i is assigned to l-th layer with full airgap. Thus, this vertex is generated only for airgap layers. Branch vertex is defined as the vertex that has multiple successors in different branches; e.g. v_2^2, v_2^4, and $\overline{v}_2^4$. Edges in E connect the vertices of adjacent wires with the direction of signal propagation. Two virtual vertices, s and t, in V_e represent source and terminal, respectively.

A weight is assigned to each edge. The edge directed to a vertex v_i^l or $\overline{v}_i^l$ has the weight of infinity if w_i cannot be asssigned to l-th layer due to wire conflict or via conflict. Otherwise, the weight is set to the reduction of setup TNS, caused by its reassigned layer and airgap insertion, where each edge weight is calculated independently of the other edges. According to the layer which w_i is assigned to, its resistance R_i and capacitance C_i are changed due to different unit resistance and unit capacitance of each layer. The layers of adjacent wires determine the number of vias between them, which change via resistance included in R_i, and the airgap insertion in w_i reduces C_i. Considering the changes in R_i and C_i, updated setup slacks for all endpoints is calculated by referring timing library and Elmore delay model [9]. The edge weight, i.e. the reduction of setup TNS, is finally obtained by summation of all slack reductions in endpoints. The weights of edges incident to s and t are set to zero.

C. Layer Reassignment with Setup Constraint (LR)

For a setup critical net, we want to find a subgraph, with the minimum sum of edge weights, on the modeled graph from where one vertex must be selected for each wire; this corresponds to the results with the largest increase in setup TNS. We adopt modified shortest path algorithm to solve it [8].

Each vertex v is associated with a cost $c(v)$ that represents the sum of edge weights in the subgraph from v to t. The costs of all vertices are initialized to infinity, except for $c(t)$ which is set to 0. For the transpose of graph, G^T, we traverse all vertices in topological order from t. For each vertex u and its successor v, $c(u)$ is added to the weight of outgoing edge to v, and $c(v)$ is updated to the value if it is smaller than $c(v)$. But for a branch vertex v, we find the minimum cost for each branch and assign the sum of them to $c(v)$. The cost at s indicates total reduction of setup TNS, which is non-positive, and each vertex on the subgraph, with the minimum cost, corresponds to the layer of each wire.

D. Airgap Insertion and Layer Reassignment with Setup and Hold Constraints (AILR)

We use A* search algorithm [10] to iteratively find subgraphs in the increasing order of setup TNS reduction, where the algorithm is modified to find a subgraph, like tree, rather than a path in the graph. The iteration terminates when the timing result of subgraph satisfies the related hold constraints.

In the graph G of hold critical net, the costs of all vertices after LR are maintained, and each edge weight $d(u, v)$ is reduced by

$$d'(u, v) = d(u, v) - c(u) + c(v), \qquad (19)$$

TABLE I
COMPARISON OF THREE INTEGRATED METHODS OF AIRGAP INSERTION AND LAYER REASSIGNMENT

Circuits	Initial circuit		LR→Restore		MIQCP			Proposed		
	TNS (ns)	WNS (ns)	ΔTNS (%)	ΔWNS (%)	ΔTNS (%)	ΔWNS (%)	Runtime (s)	ΔTNS (%)	ΔWNS (%)	Runtime (s)
s38584	32	0.27	33	9	40	11	487	40	10	14
s38417	48	0.35	17	7	25	7	1142	25	7	113
usb_funct	66	0.34	23	1	26	1	1451	26	1	230
pci	84	0.33	24	1	-	-	-	31	2	753
ac97_ctrl	172	0.57	16	0	-	-	-	25	16	2106
b18	357	0.93	20	5	28	17	2289	28	17	107
b19	501	1.03	23	2	27	17	13245	27	16	577
reed_solomon	1058	0.72	13	6	-	-	-	19	6	1690
fft_64	1405	1.04	36	4	-	-	-	43	4	1331
ethernet	1552	1.63	40	18	-	-	-	46	18	952
scdma	3635	2.08	55	6	-	-	-	59	6	870
tate_151	8227	2.95	47	9	-	-	-	57	15	10
Average			29	6				35	10	

where reduced weight $d'(u,v)$ indicates the loss in setup TNS when edge (u,v) is selected for subgraph, compared to its optimal subgraph from v to t. But for branch vertex u, $c(u)$ includes the weights from different branches, which are not in $c(v)$; thus, the minimum cost for each branch in $c(u)$ is subtracted from the weight of the edge on each branch in the calculation.

Consider Fig. 4(a), where a graph without vertices in V_f is illustrated for simplicity. The edges are associated with the reduced weights in red, and a priority queue Q is empty. A vertex s has two outgoing edges to v_1^2 and v_1^4, and each vertex set from s to v_1^2 and v_1^4 is put into Q; s is not written in the set. The set with minimum sum of weights, $\{v_1^2\}$, is extracted from Q, such that subgraph is expanded to the vertices, which are represented with red circles in Fig. 4(b). Next, we identify outgoing edges from the expanded vertex v_1^2. The vertex sets $\{v_1^2, v_2^2\}$ and $\{v_1^2, v_2^4\}$ are inserted into Q, and $\{v_1^2, v_2^4\}$ with smaller cost is extracted to expand the subgraph, as shown in Fig. 4(c). We repeat the expansion of subgraph until it reaches to t (see Fig. 4(d) and Fig. 4(e)), and the procedure is continued from the branch vertex in the subgraph, v_2^4. This procedure terminates when the subgraph is expanded to the vertices of all wires. According to the first subgraph shown in Fig. 4(f), hold slacks of related paths are updated. If there is no hold violation, the first subgraph is selected as a solution; otherwise, we search a subgraph again to find the next best solution. It starts from finding the vertex set with the minimum sum of weights from Q, which is $\{v_1^2, v_2^4, v_3^5, v_4^4\}$ with 1. $\{v_1^2, v_2^4, v_3^5, v_4^4, v_5^4\}$ is inserted into Q, and $\{v_1^2, v_2^4, v_3^5, v_4^4, v_5^2\}$ becomes the second subgraph. This iteration continues until we find the subgraph without hold violation.

Time complexity of this algorithm is $O(kNlogN)$, where N is the number of vertices and k is the number of searched subgraphs. We extract vertex set from Q for the vertices that constitute the subgraph, and it takes $O(logN)$ to delete the vertex set from Q [11]. This procedure is repeated for k subgraphs. k may become very large when the associated hold slack of net is large in negative. So we apply a pruning method for the net with the hold slack smaller than a specific value, set to -20ps in our experiments. After every iteration, we check the sum of weights for the vertex sets in Q, and the vertex sets with similar sum of weights are removed leaving only one.

IV. EXPERIMENTAL RESULTS

Experiments are carried out on a set of sequential circuits from ISCAS and ITC benchmarks as well as from OpenCores [12]–[14], which are listed in the first column of Table I. Each circuit is synthesized with modified 7-nm standard cell library, where cell delay, transition time, and gate capacitance are scaled from 28-nm industrial library [15]. Physical synthesis is performed while strict reserved routing is used; specifically, we use three metal layers (M2, M4, and M6) for horizontal routing and two metal layers (M3 and M5) for vertical routing. RC parasitics are extracted from circuit layout, where empty metal tracks are assumed to be filled with dummy wires, and they are used to estimate timing parameters, such as TNS and WNS, in static timing analysis.

After layer reassignment, circuit layout is modified according to optimization results, and if airgap is inserted in some wires, corresponding wire capacitances are scaled according to the amount of inserted airgap, while preserving the structure of RC network, in which the reduction of wire capacitance from airgap is set to 20% [1]. An exact solution is obtained by MIQCP solver [16] on 16 threads, and proposed heuristic algorithm is implemented in C++ and executed on a single thread.

A. Effectiveness of Proposed Method

Table I compares integrated methods of airgap insertion and layer reassignment in terms of TNS and WNS improvements and runtime, where WNS improvement is represented by the ratio over clock period; TNS and WNS of initial circuit, listed in columns 2-3, are used as references. LR achieves 37% and 11% improvements in TNS and WNS, respectively, on average of test circuits, which is accompanied with many hold violations. We restore the optimization of each net one by one from the net which improves the largest hold negative slack until all hold violations are removed, which is called LR→Restore. It improves TNS and WNS by 29% and 6%,

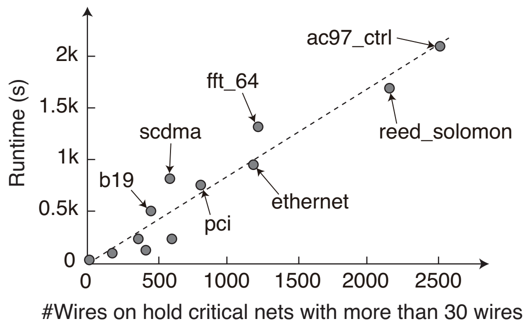


Fig. 5. Correlation between the runtime of proposed method (in y-axis) and the total number of wires on hold critical nets that consist of more than 30 wires (in x-axis).

respectively, while avoiding hold violation, which is shown in columns 4-5.

Proposed method achieves 35% and 10% improvements in TNS and WNS, respectively, as listed in columns 9-10, which is 6% and 4% larger than LR→Restore. There are a large variation in TNS and WNS improvements. It depends on the length of wires that can achieve the timing benefit from airgap. For instance, scdma with 59% TNS improvement has the average wire length of $817\mu m$ in the critical paths, while relatively short wires of $114\mu m$ are observed in reed_solomon with 19% TNS improvement. ac97_ctrl shows the largest differences in WNS and TNS improvements between LR→Restore and proposed method. LR→Restore cancels the optimizations for 370 nets out of 487 hold critical nets; on the other hand, proposed method removes all airgaps in 36 nets and employs partial airgap in 143 nets. It causes a large difference of 16ns in TNS. The same trends are observed in the most critical path which affects WNS improvement.

The improvements in TNS and WNS of MIQCP are listed in columns 6-7, where only five circuits show the results due to its heavy runtime. MIQCP calculates the optimal amount of airgap on each wire, so it is expected to be much better than proposed method that assumes the amount of airgap to be full or zero. But there is no difference between MIQCP and proposed method in terms of timing parameters because most wires in hold critical nets are short. For instance in s38584, hold critical nets occupy 19% of setup critical nets, and short wires less than $3\mu m$ are 94% of the wires in hold critical nets. The difference in the amount of airgap in those short wires has a small impact on setup timing.

B. Assessment of Runtime

The runtime of proposed method is listed in column 11 of Table I, which shows a large variation from 10 (tate_151) to 2106 seconds (ac97_ctrl). For all test circuits, most of the runtime is spent on AILR, while LR takes only a few seconds. The runtime of AILR depends on the number of wires on hold critical net, which increases the size of graph and the number of searched subgraphs. In particular, it is more influenced by the net with a large number of wires, because the number of searched subgraphs may be exponentially increased by the number of wires; for example, if a net has m wires and each wire corresponds to n vertices, the number of possible subgraphs is n^m in the worst case. Fig. 5 illustrates this relation for all test circuits in Table I, where the runtime of proposed method is represented in y-axis and the number of total wires in hold critical nets with more than 30 wires is represented in x-axis.

V. Conclusion

We have addressed airgap insertion with layer reassignment considering both setup and hold constraints. Its goal is to maximize setup TNS while assuring no hold violations. This problem has been formulated as MIQCP, and practical heuristic algorithm has been proposed whose performance is comparable to MIQCP. Experiments have demonstrated that the proposed method achieves 6% and 4% further improvements in setup TNS and setup WNS, respectively, compared to a simple approach.

VI. Acknowledgments

This work was partly supported by the National Research Foundation of Korea (NRF) of MSIT (No. 2019R1A2C2003402) and World Class 300 R&D Project of MSS (No. S2435123).

References

[1] C. Penny, S. Gates, B. Peethala, J. Lee, D. Priyadarshini, S. Nguyen, P. McLaughlin, E. Liniger, C.-K. Hu, L. Clevenger *et al.*, "Reliable airgap BEOL technology in advanced 48 nm pitch copper/ULK interconnects for substantial power and performance benefits," in *Proc. Int. Interconnect Technology Conf.*, May 2017, pp. 1–4.

[2] S. W. King, "Dielectric barrier, etch stop, and metal capping materials for state of the art and beyond metal interconnects," *ECS Journal of Solid State Science and Technology*, vol. 4, no. 1, pp. N3029–N3047, 2015.

[3] L. Wilson, "International technology roadmap for semiconductors (ITRS)," *Semiconductor Industry Association*, 2013.

[4] S. Park, S. A. B. Allen, and P. A. Kohl, "Air-gaps for high-performance on-chip interconnect part II: modeling, fabrication, and characterization," *Journal of Electronic Materials*, vol. 37, no. 10, pp. 1534–1546, 2008.

[5] L. Xia *et al.*, "Method for forming an air gap in multilevel interconnect structure," Oct. 2007, US Patent App. 11/869,409.

[6] S. Natarajan *et al.*, "A 14nm logic technology featuring 2^{nd}-generation finFET, air-gapped interconnects, self-aligned double patterning and a 0.0588 um^2 SRAM cell size," in *Proc. IEEE Electron Devices Meeting*, Dec. 2014, pp. 3.7.1–3.7.3.

[7] K. Fischer, M. Agostinelli, C. Allen, and D. Bahr, "Low-k interconnect stack with multi-layer air gap and tri-metal-insulator-metal capacitors for 14nm high volume manufacturing," in *Proc. Int. Interconnect Technology Conf.*, May 2015, pp. 5–8.

[8] D. Hyun and Y. Shin, "Integrated approach of airgap insertion for circuit timing optimization," *ACM Trans. on Des. Autom. Electron. Syst.*, vol. 24, no. 2, pp. 24:1–24:22, Feb. 2019.

[9] W. C. Elmore, "The transient response of damped linear network with particular regard to wideband amplifiers," in *Journal of Applied Physics*, vol. 19, Jan. 1948, pp. 55–63.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, July 1968.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. The MIT Press, 2009.

[12] ISCAS85. [Online]. Available: http://www.cbl.ncsu.edu/benchmarks/ISCAS85

[13] ITC99. [Online]. Available: http://www.cerc.utexas.edu/itc99-benchmarks/bench.html

[14] OpenCores. [Online]. Available: http://www.opencores.org

[15] K. Han, A. B. Kahng, H. Lee, and L. Wang, "ILP-based co-optimization of cut mask layout, dummy fill, and timing for sub-14nm BEOL technology," in *Proc. SPIE Advanced Lithography*, vol. 9635, 2015, pp. 96 350E–96 350E–14.

[16] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2015. [Online]. Available: http://www.gurobi.com